XML e rappresentazione della conoscenza per Semantic Web

Piero A. Bonatti

Dipartimento di Tecnologie dell'Informazione

Università di Milano

La ricchezza del Web

- Miniera di informazioni e servizi
- Purtroppo nè strutturati nè omogenei
  - spesso inaccessibili e/o inutilizzabili
- Di fatto, potenzialità sfruttate minimamente
  - search engines rudimentali - basati su topologia e keywords
  - l'integrazione con XML ha dei limiti
- Idea: migliorare sfruttamento risorse Web rappresentandone il *significato*

Semantic Web

Semantic Web

- Reperimento ed uso delle informazioni sulla base del loro significato
  - migliorando sia precisione che recall
- Reperimento ed uso automatici dei servizi software
  - sulla base di quello che il servizio *fa*
- La macchina deve “capire” i documenti e i servizi, in qualche misura
  - rappresentazione ed uso della conoscenza

Indice del seminario

- Ruolo dei markup languages (XML) nell'*interoperabilità* di sistemi eterogenei
  - e loro limiti
- Description logics
  - linguaggi di rappresentazione della conoscenza
  - loro rappresentazione in XML
- Service Description Logics

Sistemi distribuiti moderni

- Eterogenei, a tutti i livelli di astrazione e granularità
  - HW (workstations, palmari, cellulari, ...)
  - SW (sistemi informativi, agenti, servizi web, oggetti distribuiti,...)
  - Dati (pagine web, databases, ...)
- Componenti progettate e mantenute indipendentemente

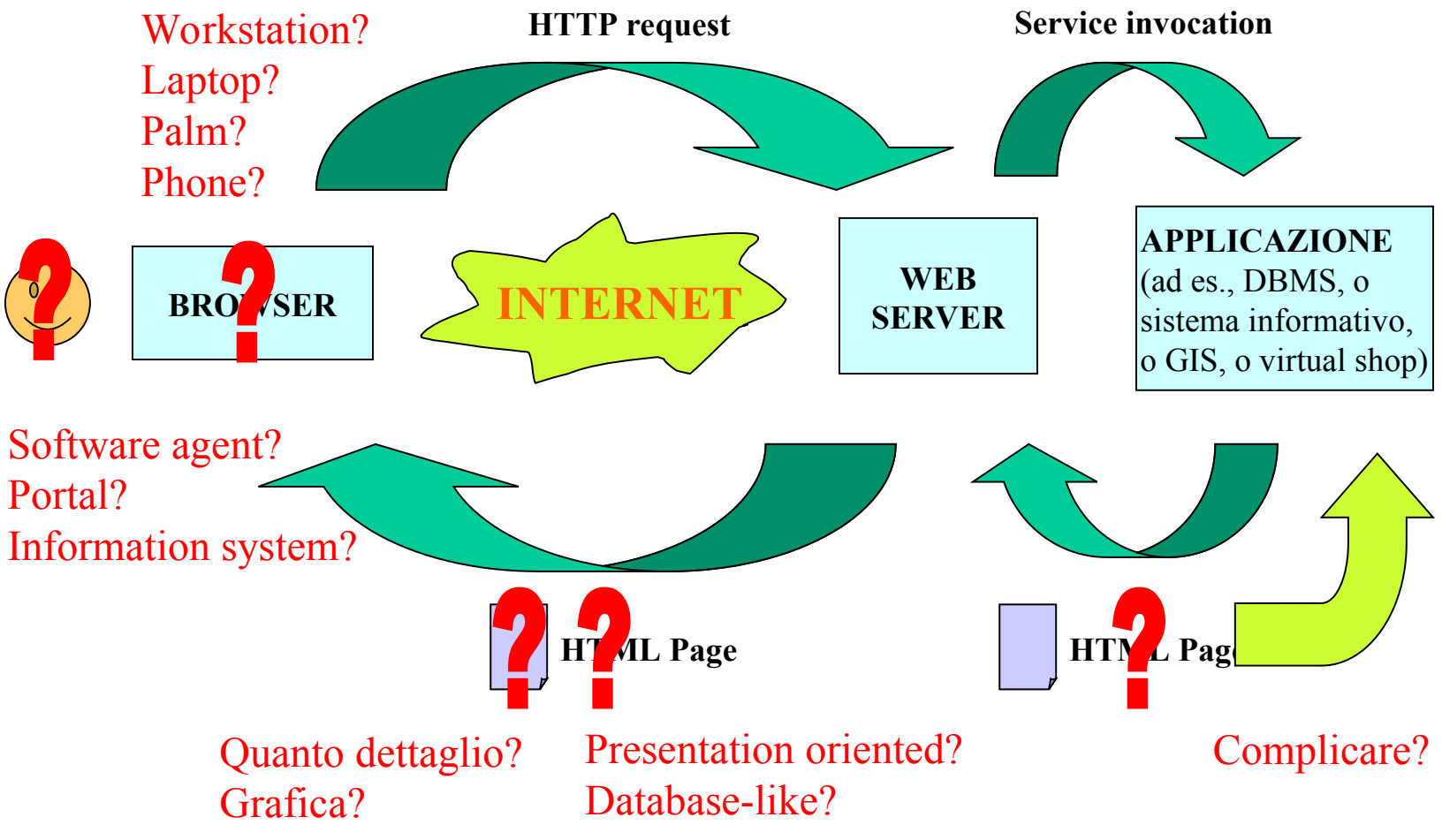
Sistemi distribuiti moderni

- Devono essere interoperabili nonostante tutto
  - ogni funzionalità del sistema può richiedere l'interazione di più componenti specializzate
- In ambiente dinamico
  - aggiunta/rimozione/sostituzione componenti
  - non noti a priori
- Cosa si vorrebbe ottenere
  - riconfigurazione spontanea del sistema
  - ...o almeno ridurre i costi degli adattamenti...

Esempio: Web applications

- Tra le caratteristiche importanti:
  - Progettazione dati e documenti senza sapere quali applicazioni li vorranno usare, ne' perche`.
  - Progettazione applicazioni senza sapere con quali altre applicazioni scambieranno dati, ne' come.
- Alcuni esempi
  - web pages & search engines
  - databases & portali
  - servers e clients per e-commerce
  - aste elettroniche & agenti

Esempio 1

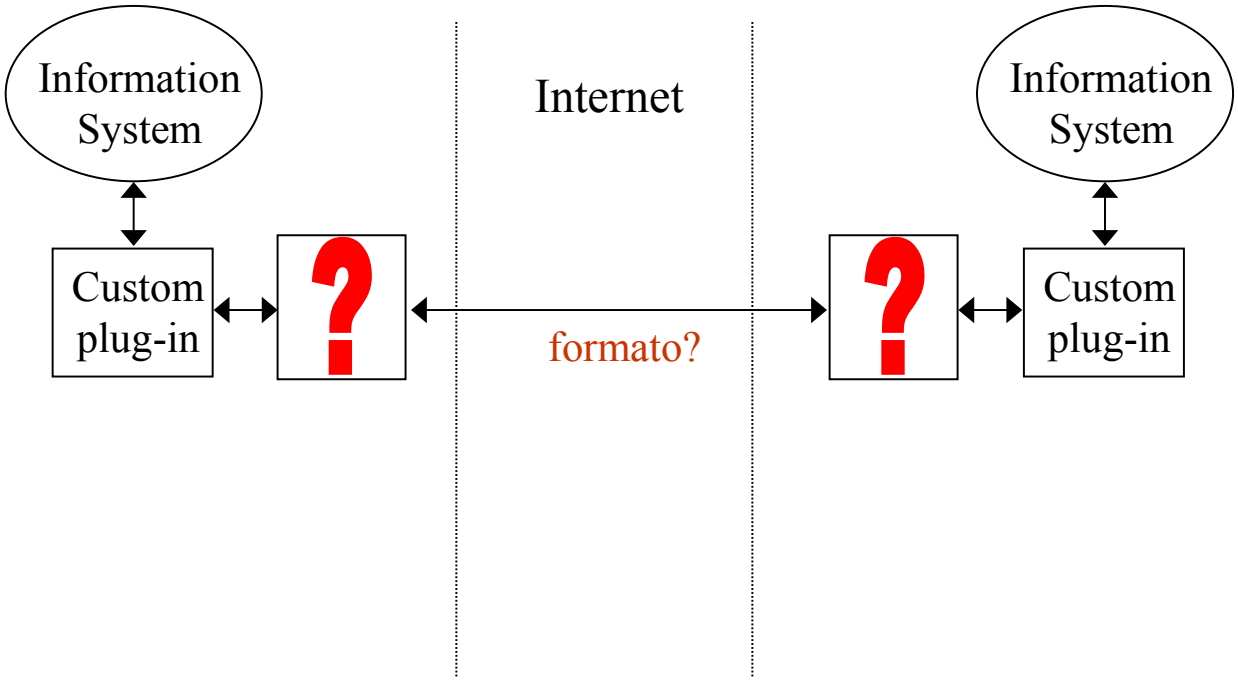

Workstation?
Laptop?
Palm?
Phone?
HTTP request
Service invocation
BROWSER
INTERNET
WEB SERVER
APPLICAZIONE
(ad es., DBMS, o sistema informativo, o GIS, o virtual shop)
Software agent?
Portal?
Information system?
HTML Page
HTML Page
Quanto dettaglio?
Grafica?
Presentation oriented?
Database-like?
Complicare?

Formati e Transazioni

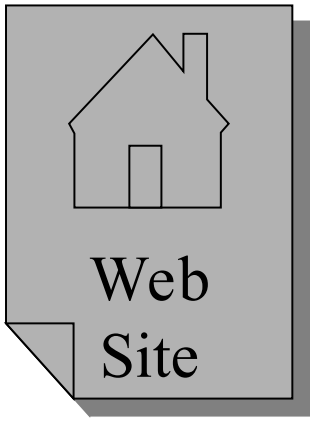
- Un problema particolare riguarda il modo in cui i dati vengono scambiati tra diverse applicazioni
- Potenzialmente ciascuna di esse usa
  - un diverso formato dei dati
  - un diverso protocollo per le transazioni (tipi di messaggi, loro significato, vincoli sulle sequenze)

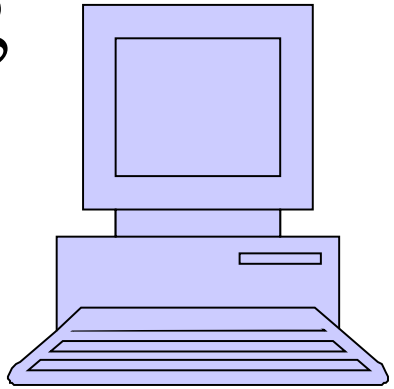
Esempio 2

- B2B (Business to Business)
  - gestione automatica degli ordini, ...
  - gli adattamenti (a nuovi partners,...) devono seguire la velocita` delle relazioni commerciali



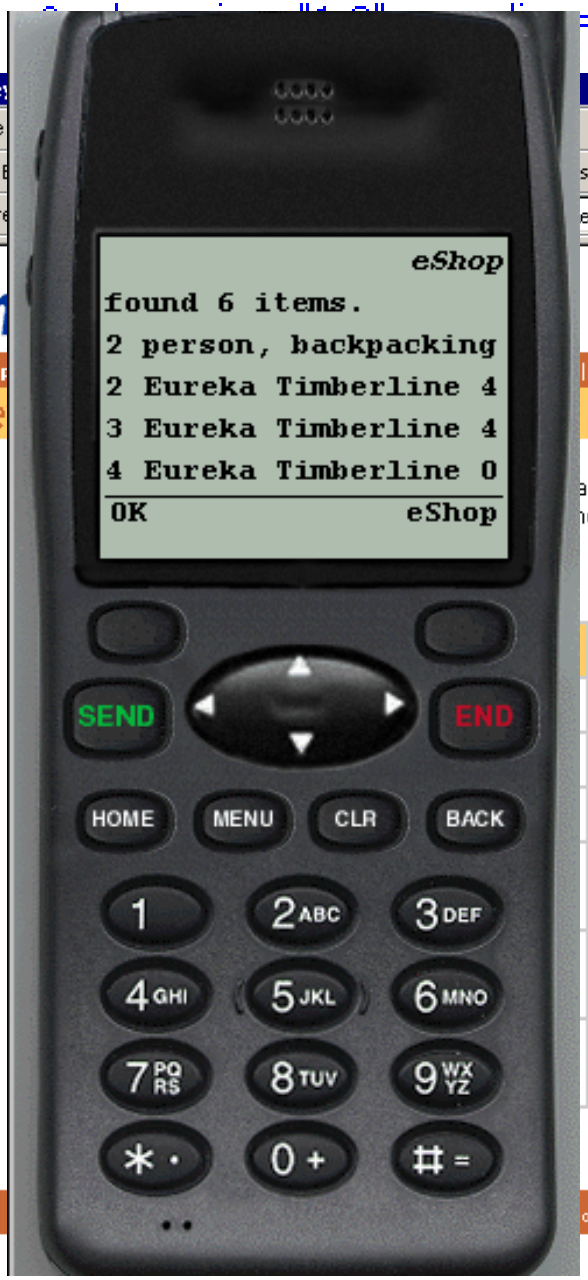
Mobile computing

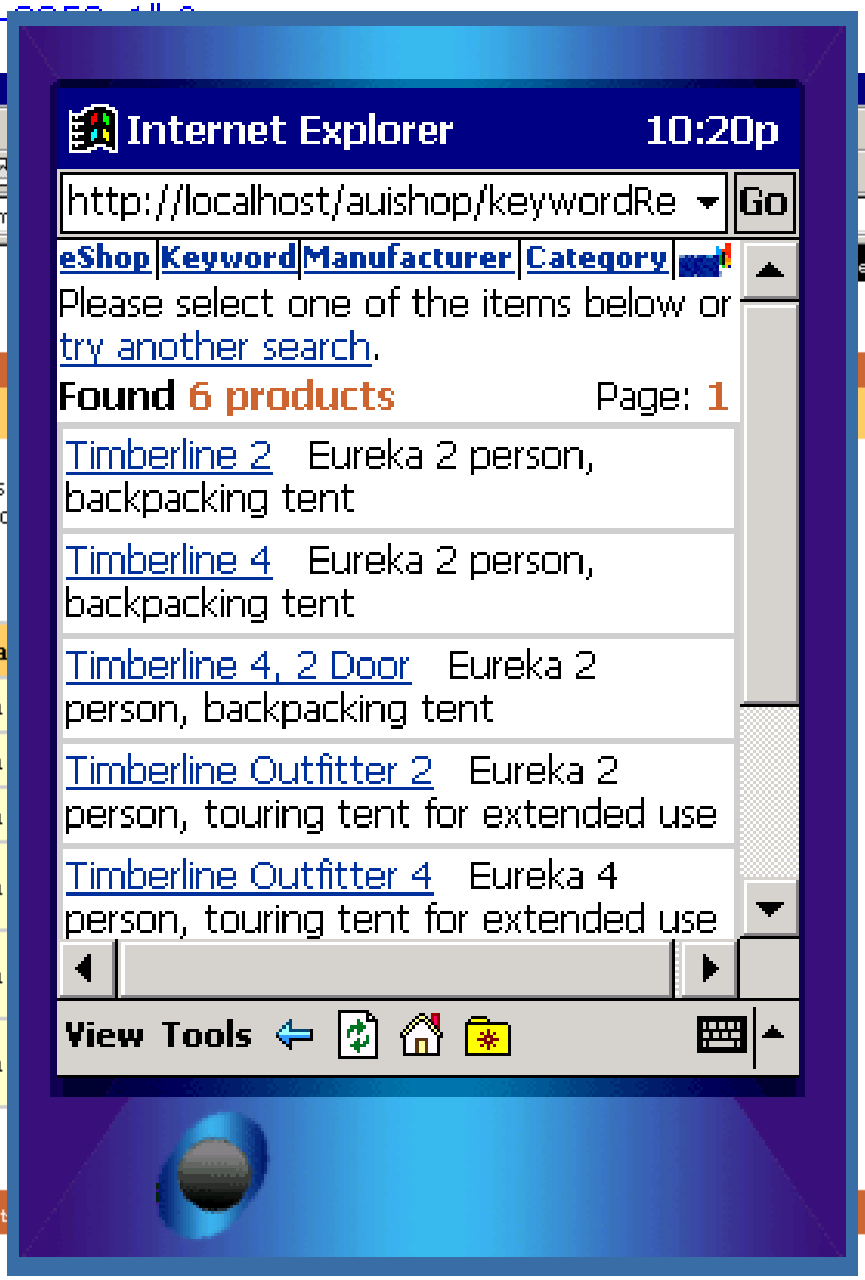




Problema

- Rendere i servizi su Internet accessibili dai terminali mobili
- Questi hanno display molto piccoli e dispositivi di input limitati
- Bisogna adattare la presentazione e le interazioni ai meccanismi di input e output forniti dai diversi tipi di terminali e ai limiti della rete wireless


eShop
found 6 items.
2 person, backpacking
2 Eureka Timberline 4
3 Eureka Timberline 4
4 Eureka Timberline 0
OK
eShop
SEND
END
HOME
MENU
CLR
BACK
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQ RS
8 TUV
9 WX YZ
* .
0 +
=


Internet Explorer
10:20p
http://localhost/auishop/keywordRe
Go
eShop
Keyword
Manufacturer
Category
Please select one of the items below or try another search.
Found 6 products
Page: 1
Timberline 2 Eureka 2 person, backpacking tent
Timberline 4 Eureka 2 person, backpacking tent
Timberline 4, 2 Door Eureka 2 person, backpacking tent
Timberline Outfitter 2 Eureka 2 person, touring tent for extended use
Timberline Outfitter 4 Eureka 4 person, touring tent for extended use
View
Tools

Strategie x flessibilità

- Disaccoppiare applicazioni da modalità uso
- Separare contenuti da presentazione
  - per facilitare ricollocazione componenti in diversi contesti
- Rappresentazione del *significato* dei dati
  - per reperirli/utilizzarli indipendentemente dal formato scelto da ciascun progettista

Orientato a rappresentazione

Filename: sample.html

<**HTML**><**BODY**>
<**H1**>A Sample Web Page</**H1**>
<**H2**>Containing a photo of a waterfall…
</**H2**>
<**P**>
<**IMG** src="picture.gif">
<**P**>
HTML consists of <**I**>standardized</**I**>
codes that are used to define the structure of information that makes up a web page.
<**P**>
HTML tags are used to place images, create hypertext links, make text bold, and more!!
</**BODY**></**HTML**>

Alcuni tags orientati ai font

- <B> e </B> sono i tag iniziale e finale per specificare un'area in grassetto (bold)
- <I> e </I> per il corsivo
- e per l'emphasized
- <FONT size=+2> e </FONT> per cambiare dimensione font

Altri tags orientati a struttura

- <H1> </H1> ... <H6> </H6> per specificare intestazioni (titoli di documenti, capitoli, sottosezioni) a piu` livelli

- Unordered lists

  <UL>

  <LI> Abc

  <LI> Def

  </UL>

Effetto:

- Abc
- Def

Da formatting a semantica

- Confrontate:

```
<p>P200 laptop
<br>Friendly Computer Shop
<br>$1438
```

con:

```
<product>
<model>P200 laptop</model>
<dealer>Friendly Computer Shop</dealer>
<price>$1438</price>
</product>
```

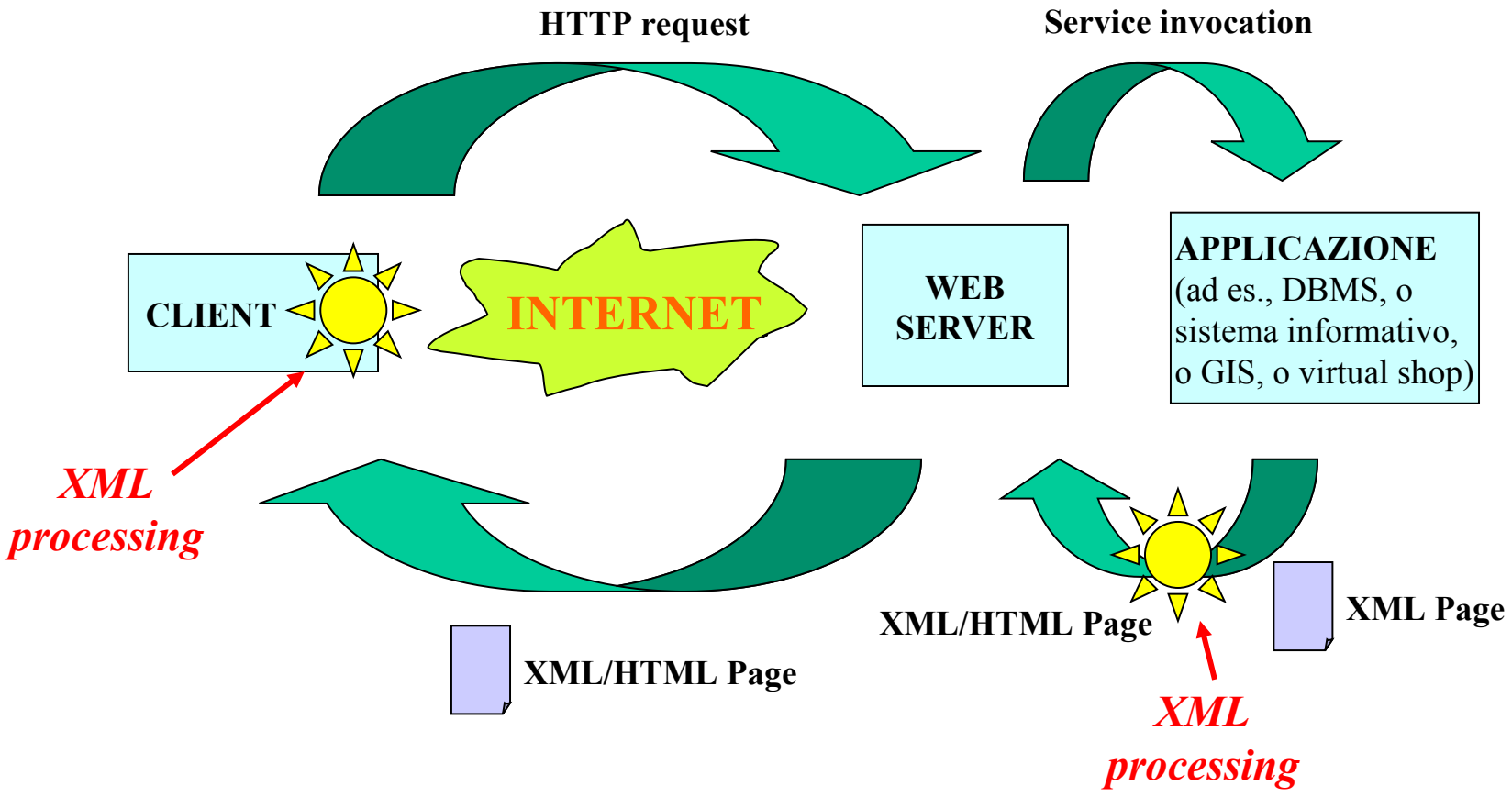
Documenti o dati?

- I tag possono giocare il ruolo dei nomi delle colonne di una tabella relazionale.
  - Quindi il documento contiene i propri metadati (si *autodescrive*).
- Sfuma la distinzione tra dati e documenti
  - documenti leggibili e visualizzabili
  - ma anche strutturati e manipolabili automaticamente (non solo per formatting)

Siamo arrivati a XML...

- Standard di rappresentazione per markup languages
- I tag e il formato possono essere definiti a piacere (DTD, XML Schemas)
- Strumenti per la manipolazione di documenti XML (Parsers, CSS, XSLT, ...)
- Rappresentazioni standard basate su XML (MathXML, WSDL, WIDL, ...)

Punti di disaccoppiamento



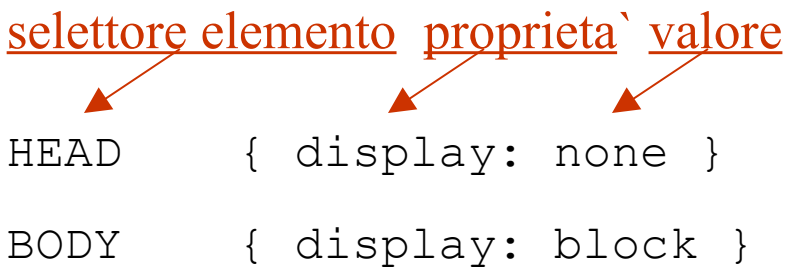
XML x database publishing

- Indipendenza database/web site design.
  - un metodo x pagine dinamiche
- Query output in XML, es.:

```
<TOYS>
<ITEM><NAME>GI John</NAME>
    <MANUFACTURER>War Toys Inc.</MANUFACTURER>
    <PRICE>50.95</PRICE>
    <IN-STOCK>3000</IN-STOCK>
<ITEM><NAME>Leggo!</NAME>
    <MANUFACTURER>Grips'R US</MANUFACTURER>
    <PRICE>64.95</PRICE>
    <IN-STOCK>2000</IN-STOCK>
 ....
</TOYS>
```

Stylesheets x presentazione

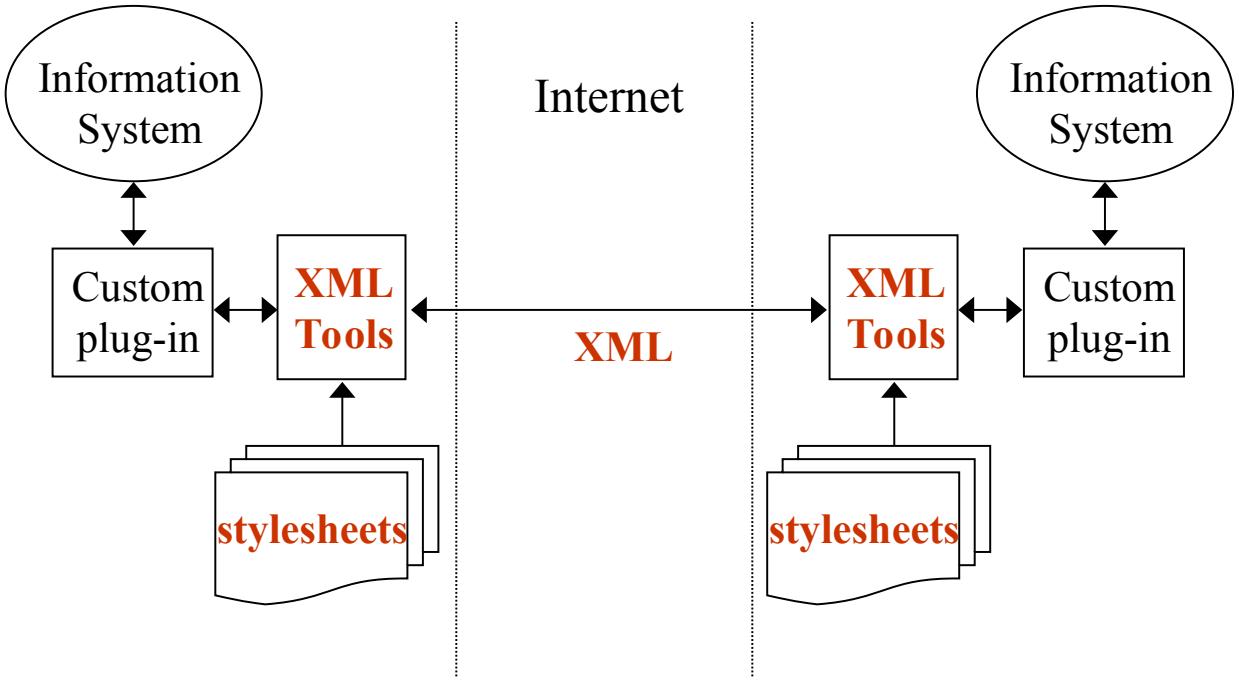
- Specifiche dichiarative di trasformazione
- CSS (Cascading Style Sheets) rule-based



- XSLT
  - linguaggio funzionale su alberi

Esempio B2B (ripreso)

- schema di interoperabilita`che dovrebbe ridurre la complessita` dei plug-in, sfruttando gli strumenti di parsing e traduzione per XML



XML x e-business

- Es.: elaborazione automatica ordini

```
<TOY-ORDER>
   <Order-No>123456</Order-No>
   <Date>20001002</Date>
   <Customer-No>98765-43</Customer-No>
 <Toy num="13245-23" quantity="12" />
 <Toy num="13425-23" quantity="21" />
 <Toy num="54321-28" quantity="15" />
 ....
</TOY-ORDER>
```

- Generati ed inviati automaticamente
- Ricevuti e serviti automaticamente
- Trasformaz. possibilmente rule-based

E-commerce: XML vs EDI

Semplificando:

- EDI: basato su interchange formats mirati a concisione (old technology): niente element names, solo posizionale.
- Tutto da rifare se cambiano formato dati o protocolli transazioni. Formati illeggibili, codice ad-hoc per visualizzazione/estrazione.
- Un'applicazione XML puo` trovare comunque i dati che le servono e ignorare il resto (dati semistrutturati). Transazioni spesso codificabili in singoli documenti. Rule-based translation

Ruolo dei Markup Languages

- Si propongono come soluzione (parziale) ai problemi di interoperabilita` tra diverse applicazioni, facilitando:
  - parsing dei formati e controlli di correttezza (in fase di creazione o trasmissione) [DTD, XML-Schemas, Parsers]
  - trasformazione da un formato ad un altro [XSLT]
  - trasformazione documenti in strutture dati direttamente manipolabili da programmi [DOM, SAX]
  - separazione formatting (visualizzazione, stampa, lettura) da contenuto del documento [XSLT]

Efficacia dei Markup Languages

- Buona per ridurre i costi dell'adattamento manuale
  - e` facile aggiungere nuovi stili di presentazione ed integrare nuovi sistemi
  - combinando stylesheets per trasformazione e presentazione si specificano concisamente numerose combinazioni predefinite
- Poco per interoperabilita` tra componenti non previsti e riconfigurazioni spontanee
  - multiagent systems, disappearing computing, sfruttamento ricchezza del Web...

Il costo dei tag semantici

- Difficolta` di standardizzazione
  - ciascuno puo` usare tag diversi, es. <prodotto> invece di <articolo>
- *Difficolta` di ricerca/riconoscimento*
- Domain specific standards
  - possono rivelarsi rigidi / statici
- Ontologie, rappres. della conoscenza
  - oggetto di ricerca, difficile ma con forti investimenti

Prospettiva

- Descrizione delle risorse Web basata sul loro significato (Semantic Web)
  - linguaggi di rappresentazione della conoscenza derivati da I.A.
  - ontologie x “spiegare” termini alla macchina
- Standard attuali o in corso di sviluppo:
  - RDF (Resource Description Format) [W3C]
    - reti semantiche
  - DAML+OIL [DARPA, OMG, ...]
    - terminological or description logics/concept languages

Meta informazione in HTML

- Proprieta` del documento, che non appartengono necessariamente al suo contenuto
- <META name=... content=... > per specificare proprieta` arbitrarie, come

```
<meta name="author" content="bonatti">
<meta name="generator" content="mozilla">
```

- Semplici associazioni attributo-valore
  - per il valore non e` prevista struttura particolare

Per illustrare RDF

User Agent Profiling

W3C / WAP

Motivazioni

- Rappresentare capacita` di elaborazione e presentazione, nonche`preferenze utente
  - per adattare la presentazione ad HW/SW
  - per filtrare i contenuti rispetto a interessi utente

Linguaggio di rappresentazione

- Standard in evoluzione basati su RDF (Resource description framework)
  - un tipo di documenti XML progettato per rappresentare proprieta` machine-understandable delle risorse su web
- Standard W3C: CC/PP (Composite Capabilities/Preference Profiles)
- Standard WAP: UAProf (User Agent Profile)

Cenni su RDF

- Elementare linguaggio di K.R.
- Il modello dei dati comprende:
  - risorse, proprieta`, statements (= subject, predicate, object), cioe`relazioni binarie (sufficienti a rappresentare relazioni arbitrarie)
- Esempio:
  - subject = “http://www.w3.org/Home/Lassila”
  - predicate = Creator
  - object = “Ora Lassila”

Elementi della sintassi RDF

- `RDF`: racchiude le descrizioni RDF
- `Description`: raccoglie il gruppo di statemen relativo ad una risorsa

namespace dei predicati

```
...xmlns:s = "some.org/some_schema/"...
<rdf:RDF>
<rdf:Description about="subject">
<s:predicate> object </s:predicate>
</rdf:Description>
...
</rdf:RDF>
```

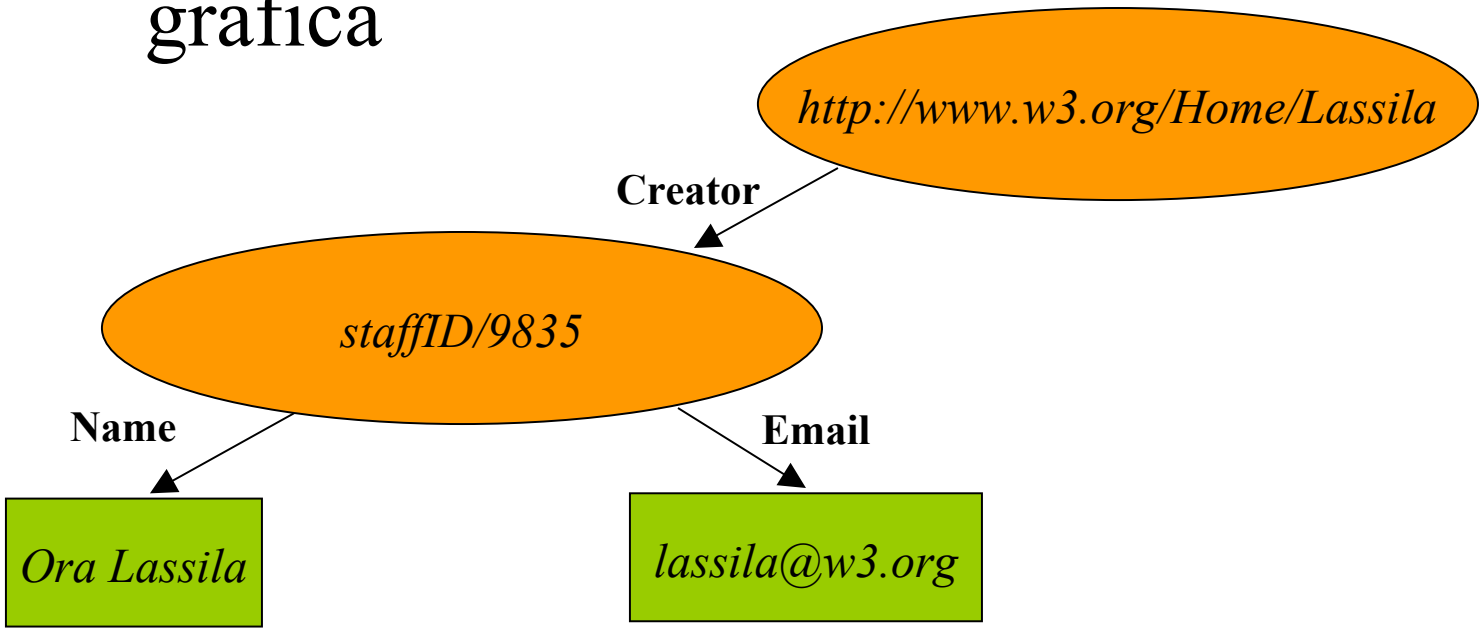
Elementi della sintassi RDF

- Le descrizioni possono usare namespace diversi ed essere innestate

```
<rdf:RDF xmlns:s = "..."  xmlns:v = "..." >
<rdf:RDF>
<rdf:Description about="http:www.w3.org/...">
   <s:Creator>
<rdf:Description about="staffID/9835">
<v:Name>Ora Lassila</v:Name>
<v:Email>lassila@w3.org</v:Email>
</rdf:Description>
   </s:Creator>
</rdf:Description>
...
</rdf:RDF>
```

Elementi di RDF

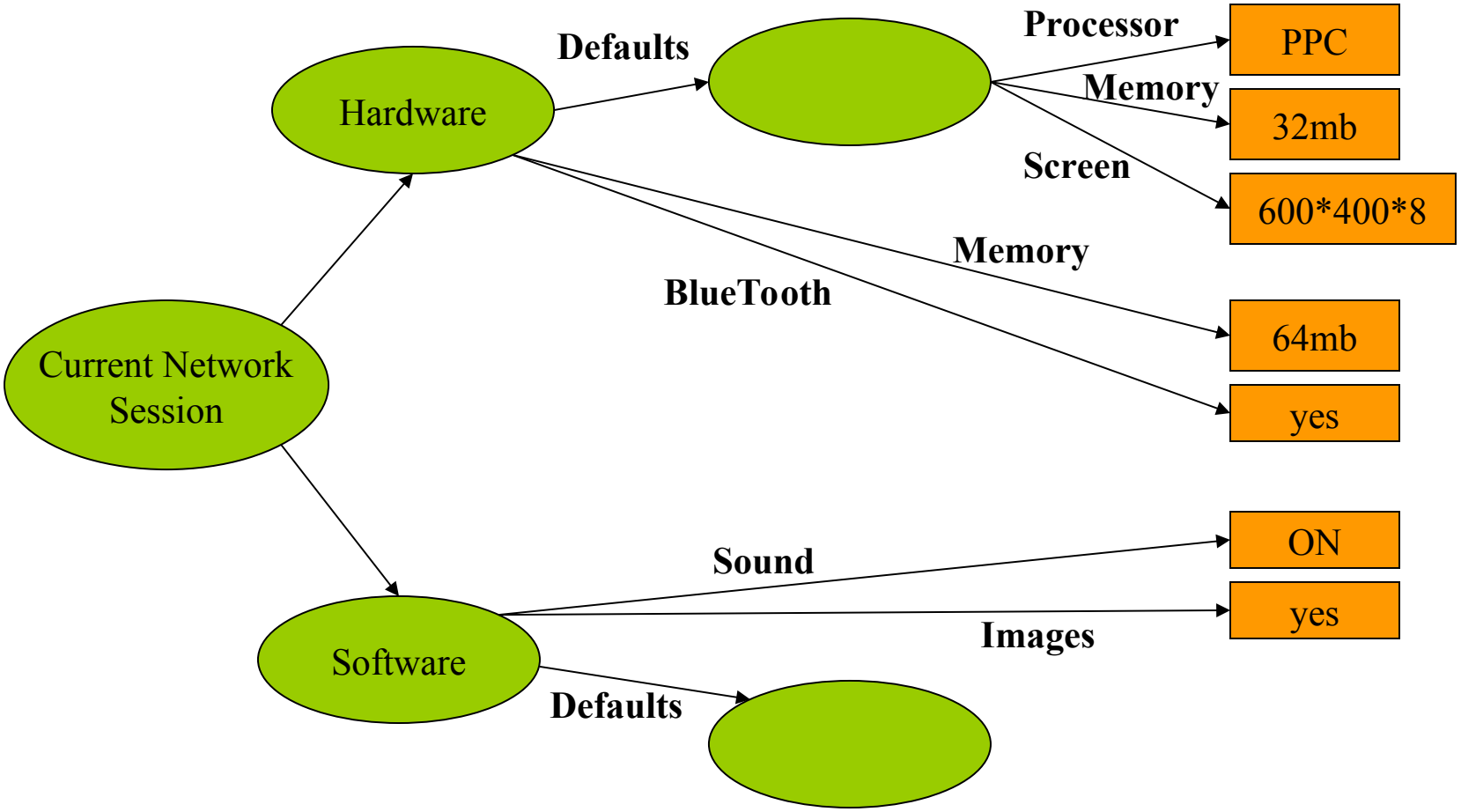
- Esistono abbreviazioni e rappresentazione grafica



Elementi di RDF

- “Containers” per rappresentare collezioni di risorse (Bag, Sequence, Alternative)
  - fanno anche le veci di classi
- Statements about statements (ad esempio chi ha inserito una certa relazione)

Proprieta` client in RDF



Rappresentazioni più complesse

- Restrizioni sui valori degli attributi
  - invece di valori specifici
- Alternative (disgiunzioni)
- Restrizioni numeriche sugli attributi
- Descrizioni parziali/incomplete
  - per pattern matching flessibile
  - per descrivere contenuti ambigui
  - ...

Esempio

```
<HTML><HEAD>
<META name="about" content="semantic_descr">
  ...
</HTML>
```

File `semantic_descr`:

```
Ontology = "rent_ontology"
rental and
        exists object (car and small)
        and
        exists price;
```

Esempio (cont.)

- Query che sfruttano l'annotazione
  - `rental and exists object (car and big)`
  - `rental and exists object car`
  - `rental and exists object car and exists price`
- Document description $\subseteq$ Query
- Per ottenere il significato desiderato occorre specificare che "big" e "small" sono incompatibili...

Esempio

```
<HTML><HEAD>
<META name="about" content="semantic_descr">
  ...
</HTML>
```

File `semantic_descr`:

rental, object, car, small, price e big sono definiti qui

```
Ontology = "rent_ontology"
rental and
        exists object (car and small)
        and
        exists price;
```

Ontologie (in informatica)

- Teorie logiche del primo ordine (insiemi di assiomi) che definiscono:
  - *concetti* (simboli di predicato unari)
  - *attributi* (simboli di predicato binari)
- cioè il loro *significato* inteso come l'insieme delle loro proprietà logiche
- Spesso espresse con *linguaggi concettuali*, con inferenza *decidibile* e talora *PTIME*

Ruolo di XML

- Permette di creare i legami tra risorse e loro significato
  - con opportuni attributi ed elementi
- di definire e fare parsing dei linguaggi delle ontologie
  - con RDF ed estensioni
- e (potenzialmente) di creare mapping tra diverse ontologie
  - strumenti di trasformazione (futuribile, limitativo)

DAML+OIL

- DAML = Darpa Agent Markup Language
- OIL = Ontology Inference Layer
- Linguaggio concettuale per la rappresentazione della conoscenza
- Sintassi definita in/con XML
- Vedremo esempi per illustrare espressività del linguaggio

Definizione classi

- Le classi definiscono concetti
  - `<daml:Class rdf:ID="Animal”/>`
- DAML estende RDF
- Altri esempi

```
<daml:Class rdf:ID="Male">
  <rdfs:subClassOf
  rdf:resource="#Animal"/>
</daml:Class>
```

Disjointness constraints

```
<daml:Class rdf:ID="Female">
  <rdfs:subClassOf
  rdf:resource="#Animal"/>
  <daml:disjointWith
  rdf:resource="#Male"/>
</daml:Class>
```

- big e small si potevano definire in modo simile

Attributi

- Relazioni binarie (oggetto, valore-attributo)

```
<daml:ObjectProperty df:ID="hasParent">
<rdfs:domain rdf:resource="#Animal"/>
<rdfs:range rdf:resource="#Animal"/>
</daml:ObjectProperty>
```

Specializzazione proprietà

- Vincoli di inclusione tra relazioni binarie

```
<daml:ObjectProperty df:ID="hasFather">
  <rdfs:subPropertyOf
  rdf:resource="#hasParent"/>

  <rdfs:range rdf:resource="#Male"/>

</daml:ObjectProperty>
```

Restrizioni su cardinalità

- I genitori sono 2

```
<daml:Class rdf:about="#Animal">
<rdfs:subClassOf>
    <daml:Restriction
  daml:cardinality="2">
  <daml:onProperty
  rdf:resource="#hasParent"/>
    </daml:Restriction>
  </rdfs:subClassOf>
</daml:Class>
```

Vincoli deboli

- Min-Max cardinality (modo per definire UniqueProperty)

```
<daml:Class rdf:about="#Person">
  <rdfs:subClassOf>
    <daml:Restriction
  daml:maxCardinality="1">
      <daml:onProperty
  rdf:resource="#hasSpouse"/>
    </daml:Restriction>
  </rdfs:subClassOf>
</daml:Class>
```

Proprietà inverse

```
<daml:ObjectProperty rdf:ID="hasChild">
  <daml:inverseOf
       rdf:resource="#hasParent"/>
</daml:ObjectProperty>
```

- e via discorrendo...

Inference engines

- Per verificare:
  - consistenza ontologie
  - sussunzione (inclusione) tra concetti (per query e classificazione automatica)
  - appartenenza a concetti/relazioni, soddisfacimento proprietà

Inference engines

- LOOM
- Classic
- FaCT
- ...

Ulteriori informazioni

- http://xml.coverpages.org/daml.html
- http://www.daml.org
- http://www.ontoknowledge.org/oil/

Non di soli dati...

- Individuazione automatica di *servizi*
- Basata su semantica (*funzionalità* del servizio)
- Per il formato con cui scambiare dati col servizio esiste standard XML: WSDL - Web Service Description Language
  - IBM, Microsoft
  - in evoluzione